Notice of Allowability	Application No.	Applicant(s)	
	09/449,021	EMMELMANN, HELMUT	
	Examiner	Art Unit	
	Chuck O. Kendall	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to *11/05/07*.

2. ☒ The allowed claim(s) is/are *1 – 6, 8, 22 – 33, 41 – 43, 51 – 96, and 114 – 128 (renumbered from 1 - 83)*.

3. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
   a) ☐ All b) ☐ Some* c) ☐ None of the:
      1. ☐ Certified copies of the priority documents have been received.
      2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
      3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

4. ☐ A SUBSTITUTE OATH OR DECLARATION must be submitted. Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL PATENT APPLICATION (PTO-152) which gives reason(s) why the oath or declaration is deficient.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   (a) ☐ including changes required by the Notice of Draftsperson's Patent Drawing Review (PTO-948) attached
      1) ☐ hereto or 2) ☐ to Paper No./Mail Date _____.
   (b) ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)
2. ☐ Notice of Draftperson's Patent Drawing Review (PTO-948)
3. ☐ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date _____
4. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material
5. ☐ Notice of Informal Patent Application
6. ☒ Interview Summary (PTO-413), Paper No./Mail Date *01/24/08* .
7. ☒ Examiner's Amendment/Comment
8. ☒ Examiner's Statement of Reasons for Allowance
9. ☐ Other _____.

Examiners Amendment

1. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this examiner's amendment was given in a telephone interview with Richard Nebb registration no. *33,540* on January 24th 2008. Claims are being amended to clarify claim limitations.

The proposed amendment dated 1/24/08 has been accepted and adopted by the Examiner-see attachment herein.

The application has been amended as follows:

IN THE CLAIMS

2. Please amend claims 1 – 6, 8, 22 – 33, 41 – 43, 51 – 96, and 114 – 128 as attached hereto see pages 5 – 28.

See attached document as proposed by Applicant to amend claims

Reasons for Allowance

3. Examiner has reviewed and considered Applicant's arguments as indicated in Applicant's proposed amendment of 11/05/07 which is being adopted into the record as an examiner's amendment and per Applicants arguments on page 2 of his 11/05/07 response and per Interview Summary of 07/13/07 and 01/24/08, claims 1 – 6, 8, 22 – 33, 41 – 43, 51 – 96, and 114 – 128 are now in condition for allowance.

The following is an Examiner's statement of reasons for allowance.

The prior art of record does not teach or fairly suggest at least the limitations of:

"...a document generator program running at least part of one of the applications being edited and generating the generated documents, said generated documents including additional editing features for interpretation by the browser program; and

an editor dynamically operating on the generated documents displayed by the browser program via editing features..." , and as best illustrated by FIG. 18, in such a manner as recited in independent claims 1, 22, 59, 74,114, 125.

"...a plurality of components residing in the data store, including at least one selected component executing instructions contained in said selected component on the server computer...a component editor controlled by a fourth software program, said fourth software program providing instructions for dynamically editing selected components on a page template...", and as best illustrated by FIG. 18, in such a manner as recited in independent claim 6.

"...a first software program including instructions for transforming at least one first document retrieved from the document store into a second document having features which permit dynamic editing of the first document such that at least part of the second document appears and functions similar to the first document...", and as best illustrated by FIG. 18, in such a manner as recited in independent claim 26.

"...the second software program transmitting, while processing selected requests, second documents to the first software program that make the first software program display a user interface for editing functions used for maintaining components on document templates, and

a third software program used by the second software program while processing selected documents requests, the third software program including third instructions for dynamically modifying document templates in order to dynamically perform said editing functions..." , and as best illustrated by FIG. 18, in such a manner as recited in independent claim 51.

"...the editor program having instructions for allowing the user to dynamically edit at one document displayed by the browser on a display device, wherein scripts contained in said document remain running during editing, the editor program including a first software program for execution within the browser having instructions for processing selected clicks on the view of said document displayed in the browser by initiating editing functions...." , and as best illustrated by FIG. 18, in such a manner as recited in independent claim 90.

Therefore, all claims remaining claims 1 – 6, 8, 22 – 33, 41 – 43, 51 – 96, and 114 – 128 are in condition for allowance.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Correspondence Information

4. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Chuck Kendall whose telephone number is 571-2723698. The examiner can normally be reached on 10:00 am - 6:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on 571-2723695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/Chuck O Kendall/
Examiner, Art Unit 2192

CLAIM AMENDMENTS:

1. (currently amended) A computer-readable medium encoded with computer programs having executable instructions ~~software development system~~ for editing software applications that run on a data network which couples a server computer and a client computer, wherein the client computer runs a browser program, and whereupon request by the browser program, ~~an~~ at least one of the applications generates generated documents for display by the browser program on a display device and responds to the request with the generated documents ~~pages~~, comprising:

a ~~page~~ document generator program running at least part of one of the applications being edited ~~developed by~~ and generating the generated documents, said generated documents including additional editing features for interpretation by the browser program; and

an editor program ~~directly~~ dynamically operating on the generated documents ~~pages~~ displayed by the browser program via the editing features~~, thereby allowing the user to work on a functional application during development~~.

2. (currently amended) A computer-readable medium ~~software development system~~ as ~~claimed~~ in claim 1, further encoded with ~~further comprising~~ a plurality of components, and wherein ~~developed~~ the software applications comprise at least one ~~page~~ document template capable of containing components, and wherein the editor provides features to insert, modify and delete a component on at least one ~~page~~ document template, and wherein the ~~page~~ document generator executes selected components on ~~page~~ document templates.

3. (currently amended) A computer-readable medium ~~software development system~~ as ~~claimed~~ in claim 2, wherein at least one of the components reacts interactively on user input by executing instructions of said component on the server computer.

4. (currently amended) A computer-readable medium ~~software development system~~ as in claim 3, wherein at least one of the components contains at least one other component.

5. (currently amended) A computer-readable medium ~~software development system~~ as in claim 3, wherein the set of components on ~~pages~~ documents generated from at least one ~~page~~ document template can vary for different requests of said ~~page~~ document template.

6. (currently amended) A software development system for use in a data network which couples a server computer to a client computer, wherein the client computer includes a first software program for generating a request for one or more pages from the server computer and for displaying pages on a display device, and wherein the server computer includes a second software program for receiving and processing the request from the client computer, for generating and storing pages, and for transmitting pages to the client computer in response to requests, the server computer further comprising:

a data store,

a plurality of components residing in the data store, including at least one selected component ~~that reacts interactively on user input by~~ executing instructions contained in said selected component on the server computer;

a plurality of page templates residing in the data store, at least one page template having at least one selected component incorporated therein; ~~and~~

a server processor controlled by a third software program, said program providing instructions for selecting a page template based on the request from the client computer and instructions for generating a page from the page template for transmission to the client computer; and

a component editor controlled by a fourth software program, said fourth software program providing instructions for dynamically editing selected components on a page template.

7. (cancelled).

8. (original) The development system of claim 6, wherein a component is nested within a component.

9. (withdrawn) A method for generating documents for display by a browser using components that react interactively on user input by executing instructions on a server, comprising the following steps for execution on the server upon a document request:

assigning a unique identifier to at least one of the components; and

embedding the unique identifier into a generated document.

10. (withdrawn) The method of claim 9, further comprising storing data on the server representing at least one of the components.

11. (withdrawn) The method of claim 10, further comprising:

analyzing the request sent by the browser for unique identifiers; and

calling a function for the interactive components referenced by at least one of the unique identifiers contained in the request.

12. (withdrawn) The method of claim 11, wherein at least one of the components is contained on a document template.

13. (withdrawn) The method of claim 11, wherein at least one of the components is called by a program.

14. (withdrawn) The method of claim 11, wherein at least one of the components is called by a another component.

15. (withdrawn) The method of claim 11, wherein the data is stored into an object of an object oriented programming language and wherein the function is a method of the object.

16. (withdrawn) A method for implementing client server applications, comprising:

storing data objects on a server and assigning a unique identifier to each data object;

dynamically generating a document with the unique identifier embedded in the document; and

analyzing requests for unique identifiers and calling at least one function for a data object associated with one of the unique identifiers found in the request.

17. (withdrawn) The method of claim 16, wherein the unique identifier is embedded inside a uniform resource locator contained in a tag of the document.

18. (withdrawn) The method of claim 16, wherein the unique identifier is embedded in scripts contained in the document.

19. (withdrawn) The method of claim 16, wherein the unique identifier is unique within a single session.

20. (withdrawn) The method of claim 16, wherein the unique identifier is unique within all documents generated by a single server within a defined time period.

21. (withdrawn) The method of claim 16, wherein the data objects are created by an object-oriented programming language and said function is a method of one of these objects.

22. (currently amended) A computer-readable medium encoded with computer programs having executable instructions ~~computer running an application~~ to ~~develop~~ edit and maintain applications using a web browser, comprising:

an editor program operable within the web browser and having instructions for dynamically inserting, deleting, and modifying components on document templates; and

a document generator program having instructions for processing document templates, for executing components, and for generating documents from the document templates that are understandable by the web browser.

23. (currently amended) A computer readable medium as in claim 22, wherein the editor program operates functional applications in an edit mode permitting editing directly in the web browser.

24. (currently amended) A computer readable medium as in claim 23 wherein at least one of the components contains instructions and can react on subsequent document requests containing user responses by executing selected instructions of said component.

25. (currently amended) A computer readable medium as in claim 24~~, wherein the computer further~~ encoded with ~~comprises~~:

~~a store of~~ component classes, each component class implementing one component kind; and

a parser program able to detect components marked on document templates;

wherein the document generator program works upon a document request using component classes to generate browser code; and

wherein the editor program is capable of showing a menu of components for insertion into the document templates.

26. (currently amended) A system having a data network which couples a server computer to a client computer, the server computer running an application to modify documents on ~~a~~ the server computer~~ in a data network which couples said server computer to a client computer~~, the server computer comprising:

a document store;

a first software program including instructions for transforming at least one first document retrieved from the document store into a second document having features which permit dynamic editing of the first document such that at least a part of the second document appears and functions similar to the first document; and

a second software program including instructions to receive information from the client computer and instructions to modify the first document stored in the document store.

27. (previously amended) The system of claim 26, wherein the first document includes at least one component being executed by the first software program .

28. (previously amended) The system of claim 27, wherein the second document includes handles and choosing one of the handles selects a component for an editing operation.

29. (previously amended) The system of claim 28, wherein at least one handle indicates the position of at least one component contained in the first document and said editing operation includes modifying the component, deleting the component, and displaying information regarding the component.

30. (previously amended) The system of claim 26, wherein the features include scripts.

31. (previously amended) The system of claim 30, wherein the scripts encapsulate information from the first document.

32. (original) The system as in claim 26, wherein the features incorporate information regarding the first document into the second document.

33. (currently amended) A system as in claim 32, wherein the information incorporated into the second document is used on the client computer in order to send change requests for the first document to the server computer.

34. (withdrawn) A method for generating a document for display in a browser from a document template containing components, comprising:

for each component denoted on the document template, identifying a component class of the component; and

based on data contained in a request initiated by the browser storing a first object of the component class, the first object representing the component.

35. (withdrawn) The method of claim 44, further comprising calling a metho said component class to generate browser code, said method being the constructor.

36. (withdrawn) The method of claim 34, further comprising, for all compone having a name attribute, looking up the component object in session memory based on said name attribute.

37. (withdrawn) The method of claim 34, further comprising, for at least one component kind, for all components denoted on the document template having said component kind;

generating a unique identifier;

assigning said unique identifier to said object, and

embedding said unique identifier into the browser code.

38. (withdrawn) The method of claim 37, further comprising:

inserting objects for the components of at least said component kind into a list listening components;

working through all objects stored in the list of listening components whose unique identifier occurs inside a name in the form data set; and

calling a method of at least one of these objects.

39. (withdrawn) The method of claim 34, wherein the document template is parsed into a list of nodes, including text and component nodes, said method further comprising:

determining if the current node is text or a component;

if component, then calling a method for the component, comprising:

evaluating the attributes of the component if necessary;

identifying the component class associated with the component; and

calling the constructor method of the component class,

said constructor method generating browser code;

if text, then generating the text; and

repeating these steps for each node.

40. (withdrawn) The method of claim 39, wherein at least one component contains nested components and the method of claim 39 is recursively performed for all nodes nested inside the component.

41. (currently amended) A ~~software development system~~ computer-readable medium as in claim 1, the editor program comprising a client part for execution on the client computer.

42. (currently amended) A ~~software development system~~ computer-readable medium as in claim 41, wherein the client part comprises instructions for execution during editing that are automatically downloaded from the server computer in a request prior to editing.

43. (currently amended) A ~~system~~ computer-readable medium as in claim 26, additionally comprising at least one script for automatic download to the client that works in cooperation with the second document to permit editing of the first document.

44. (withdrawn) The method of claim 34 wherein storing the first object comprises creating a new object as necessary.

45. (withdrawn) The method as in claim 34 wherein components are denoted on document templates using tag syntax.

46. (withdrawn) The method as in claim 45 wherein the tag name identifies a component class.

47. (withdrawn) The method as in claim 36 wherein components are denoted on document templates using tag syntax, wherein the tag name identifies a component class.

48. (withdrawn) The method as in claim 36 wherein the component object, if found, is reused to store the first object.

49. (withdrawn) The method as in claim 36 wherein in case a component has a name attribute but no component object is found a new object is created and stored under said name in session memory.

50. (withdrawn) The method as in claim 49 wherein new objects are created for all components not having a name attribute.

51. (currently amended) A system having at least one computer running a second software program for editing components on web document templates for use with a first software program including first instructions for generating a document request to obtain at least one generated document from the ~~a~~ second software program and for displaying the generated document, the second software program capable of receiving and processing the document request and of transmitting first documents to the first software program in response to requests, said system comprising:

a plurality of components containing instructions to generate browser code,

a plurality of document templates,

the second software program transmitting, while processing selected requests, second documents to the first software program that make the first software program display a user interface for editing functions used for maintaining components on document templates, and

a third software program used by the second software program while processing selected document requests, the third software program including third instructions for dynamically modifying document templates in order to dynamically perform said editing functions.

52. (previously amended) The system of claim 51, wherein at least some components include fourth program instructions including steps to generate browser code

for transmission to the first software program in response to a request from the first software program, wherein the browser code can differ for multiple requests for the same document template.

53. (currently amended) A system as in claim 52 ~~running on~~ having a data network, coupling the ~~a server~~ computer and a client computer, the first program running on the client computer~~, the second program running on the server computer~~.

54. (currently amended) A system as in claim 52 wherein second documents include HTML pages with embedded scripts.

55. (currently amended) The system of claim 52, wherein the editing functions includes adding a component to a document template, removing a component from a document template, and modifying a component on a document template.

56. (previously added) The system of claim 52, further comprising a fifth software program used by the second software program while processing selected document requests, the fifth software program including fifth instructions for generating generated documents from document templates thereby calling fourth program instructions.

57. (previously added) The system of claim 56, wherein the generated document includes, if requested in edit mode, edit features for interpretation by the first software program.

58. (previously amended) The system of claim 56 further comprising instructions to allow the user to click on the generated document to select items to perform edit functions on.

59. (currently amended) A software development system having at least one computer running an application for developing dynamic web documents, said dynamic

web documents operating by being transformed into an end user's view upon a request by a web browser, the end user's view being provided to the browser for display on a display device in response to the request, comprising:

an editor program having instructions for dynamically editing dynamic web documents,

a document generator program having instructions for generating generated documents from dynamic web documents which look and function similar to the end user's view of the documents,

the editor program comprising first instructions for requesting that the document generator program ~~to~~ processes a dynamic web document during editing ~~leading to~~ thereby resulting in a generated document,

the system ~~further~~ comprising second instructions for displaying at least some information items contained on said generated document in a view which allows the user to select an item to which a modification function will be applied,

the editor program ~~further~~ comprising third instructions to modify the dynamic web document to perform said modification function.

60. (currently amended) The software development system of Claim 59 ~~running on~~ comprising a data network which couples a server computer and a client computer, the document generator program running on the server computer, the editor program at least partly running on the client computer.

61. (currently amended) The software development system of claim 60 ~~further~~ comprising fourth instructions for execution during document generation to collect edit-information for use by the editor program.

62. (currently amended) The software development system of claim 60, wherein the editor program uses a web browser for displaying said view.

63. (currently amended) The software development system of claim 60, comprising instructions for ~~able to~~ automatically repeating the request~~ing~~ that the document generator ~~to~~ processes the dynamic web document when ~~if~~ required.

64. (currently amended) The software development system of Claim 59 further comprising a plurality of components including at least one component marked on said dynamic web document and including instructions for use by the document generator program to generate browser code.

65. (currently amended) The software development system of claim 64, wherein the editor program uses a web browser for displaying said view.

66. (currently amended) The software development system of claim 64, wherein the modification function~~s~~ includes insertion of a component, ~~delete~~ deletion of a component, and ~~modify~~ modification of a component.

67. (previously added) The software development system of claim 59, wherein said view looks, except for editing features, similar to the end-user view of the generated document.

68. (currently amended) The software development system of claim 59 ~~further~~ comprising sixth instructions to collect edit-information for use by the editor program, said sixth instructions for execution during document generation.

69. (currently amended) The software development system of claim 68, wherein the editor program uses the edit-information to correctly modify the dynamic web document.

70. (currently amended) The software development system of claim 69, further comprising a plurality of components wherein the edit-information comprises position information on selected components marked on the [[-]] dynamic web document.

71. (currently amended) The software development system of claim 59, wherein the editor program uses a web browser for displaying said view.

72. (currently amended) The software development system of claim 71, wherein the first instructions comprise seventh instructions for initiating a reload in the browser.

73. (currently amended) The software development system of claim 59 wherein the editor program ~~further~~ comprises eighth instructions to display information on at least one element of at least one dynamic web document, that is replaced during document generation, without requesting that the document generator program ~~to~~ generates a document.

74. (currently amended) A ~~software development~~ system having at least one computer running an application for developing document templates that are intended for transformation into generated documents for display by a first software program, the first software program including first instructions for generating a document request to obtain at least one generated document and for displaying the generated document on a display device, comprising:

a plurality of components~~, that include~~ having instructions to generate browser code for transmission to the first software program,

an editor program having instructions for dynamically ~~capable of~~ performing editing functions to maintain~~ing~~ components on document templates, the components having the ability ~~capable~~ to cooperate with the editor,

a plurality of document templates having said components denoted thereon, and

a document generator program having ~~comprising second~~ instructions to, upon document requests, generate generated documents from at least one document template for display by the first software program wherein the set of components on the generated documents can vary for different document requests for said document template.

75. (currently amended) The ~~software development~~ system as in claim 74, wherein the editing functions comprises adding a component, modifying a component, and deleting a component.

76. (currently amended) The ~~software development~~ system as in claim 74, wherein tag syntax is used to denote at least one component on at least one document template, whereby the tag name identifies the component kind.

77. (currently amended) The ~~software development~~ system of claim 74 ~~running on a data network, which couples~~ comprising a server computer coupled to ~~and~~ a client computer by a data network, the document generator program running on the server computer, and the editor program running, at least partly, on the client computer.

78. (currently amended) The ~~software development~~ system as in claim 74, wherein at least one component~~,~~ that can react interactively on subsequent document requests~~,~~ can be excluded from said generated document upon selected document requests for said document template~~, be excluded from said generated document~~.

79. (currently amended) The ~~software development~~ system as in claim 78 ~~further~~ comprising third instructions to prevent excluded components from reacting on subsequent document requests.

80. (currently amended) A ~~software development~~ system as in claim 79, said third instructions comprising fourth instructions to, upon a first document request, store information in session memory on some of the components[[~~,~~]] that are present on the document generated based on the first document request, and fifth instructions to, upon subsequent document requests, only react on components that have been remembered in session memory thereby avoiding tampering with excluded components on the side of the first program.

81. (currently amended) A ~~software development~~ system as in claim 74 wherein at least one first component contains sixth instructions to decide upon a request for said document template about exclusion of components nested inside the first component from the generated document.

82. (currently amended) A ~~software development~~ system as in claim 74 the system ~~able to provide~~ providing an editable view taking the varying set of components into account.

83. (currently amended) A ~~software development~~ system as in claim 74 ~~the system~~ ~~able to provide~~ providing an editable view that includes and excludes selected components on different requests for said document template ~~similarly as~~ similar to the end user's view of the document template.

84. (currently amended) A ~~software development~~ system as in claim 74 wherein a document generated for at least one document template contains more components than the document template for at least one document request.

85. (currently amended) The ~~software development~~ system as in claim 74, wherein multiple instances of at least one third component denoted on the document template can be included in at least one of the documents generated from said document template.

86. (currently amended) The ~~software development~~ system as in claim 74, ~~further~~ comprising seventh instructions to assign a unique identifier to each component instance of at least one seventh component, whereby the seventh component includes eighth instructions to qualify names generated into the browser code with the unique identifier.

87. (currently amended) A ~~software development~~ system as in claim 74, wherein at least one fourth component contains ninth instructions to decide upon a request about

how many instances of components nested inside the fourth component are included in the documents generated from said document template.

88. (currently amended) A ~~software development~~ system as in claim 74 the editor program able to provide an editable view that includes multiple instances of selected components similar~~ly as~~ to the end user's view of the document template.

89. (currently amended) A ~~software development~~ system as in claim 74 wherein at least one sixth component includes tenth instructions to display the sixth component, the tenth instructions being used to generate browser code for displaying the sixth component during editing as well as during normal use of the component.

90. (currently amended) A system having at least one computer running an ~~An~~ editor program for use with a web browser, the editor program having instructions for allowing the user to dynamically edit at least one document displayed by the browser on a display device, wherein scripts contained in said document remain ~~functional~~ running during editing, the editor program including a first software program for execution within the browser ~~and~~ having instructions for processing selected clicks on the view of said document displayed in the browser by initiating editing functions.

91. (currently amended) The system ~~editor~~ as in claim 90 ~~using~~ wherein the editor program includes instructions to display at least two windows, a first browser window displaying said document and a second window for displaying information on an element contained in said document.

92. (currently amended) The system as ~~editor~~ in claim 90 ~~further~~ comprising a second software program having instructions for storing modifications on said document in cooperation with the first software program.

93. (currently amended) The system ~~editor~~ as in claim 92 further comprising a third program having instructions for transforming an original document into the

document, the browser displaying the document as said view looking similar to the original document and interpreting editing features contained in the document.

94. (currently amended) The system ~~editor~~ as in claim 93 wherein said original document is a dynamic document having components denoted thereon, the third software program ~~further~~ comprising instructions for generating browser code in cooperation with selected instructions contained in the components.

95. (currently amended) The system ~~editor~~ as in claim 94 comprising a client computer connected to a server computer via a data network, wherein the browser together with the first software program is running on ~~a~~the client computer ~~connected to a server computer via a data network, wherein~~, and the second and the third software program run on the server computer.

96. (currently amended) The system ~~editor~~ as in claim 90 wherein links contained in said document stay functional allowing the user to browse and edit at the same time.

97. (withdrawn) A system for displaying dynamically generated documents in a data network coupling a server computer to a client computer, wherein the client computer has a first software program including first instructions for generating a document request to obtain at least one generated document from the server computer and for displaying the generated document, comprising:

a plurality of components for execution on the server computer, including a first component including second program instructions to generate browser code and third program instructions for execution on the server which are initiated by the user interacting with the first component, and,

fourth program instructions on the server computer for, based on data contained in a document request initiated by the first software program on the client computer, generating generated documents for transfer to the client computer and display by the first software program, thereby calling second program instructions of components.

98. (withdrawn) The system of claim 97, the server computer further comprising fifth program instructions for analyzing said data and for calling third program instructions of first component as necessary.

99. (withdrawn) The system of claim 98, further comprising a plurality of document templates residing in the data store, at least one of the document templates having at least one second component denoted thereon.

100. (withdrawn) The system of claim 99, wherein tag syntax is used to denote the second component, whereby the tag name identifies a component.

101. (withdrawn) The system of claim 99, wherein at least one third component denoted on a document template contains the first component.

102. (withdrawn) The system of claim 101, wherein third components implementation scheme includes logic to decide how often to insert the first component into the generated document.

103. (withdrawn) The system of claim 98, the server computer further comprising sixth program instructions for, based on the document request, deciding to insert more than one instance of the first component into the generated document.

104. (withdrawn) The system of claim 103, making sure that multiple instances of the first component do not interfere by qualifying names generated into the browser code using unique identifiers.

105. (withdrawn) The system of claim 98, the server computer further comprising seventh program instructions for, based on the data, deciding to exclude the first component from the generated document.

106. (withdrawn) The system of claim 105, wherein fifth instructions call third instructions only if the first component was contained on a page previously transferred to the client.

107. (withdrawn) The system of claim 98, wherein fifth program instructions include eighth program instructions to analyze said data for user interactions with multiple components and to call third program instructions of multiple components as necessary.

108. (withdrawn) The system of claim 107, wherein components include ninth instructions to check for errors and fifth instructions include tenth instructions to call ninth instructions of components as necessary and to suppress subsequent calling of third instructions in case of errors.

109. (withdrawn) The system of claim 98, further comprising eleventh program instructions for storing a data object in session memory representing at least one component instance included in a generated document, said eleventh program instructions for execution while dynamically generating a document.

110. (withdrawn) The system of claim 109, wherein third program instructions are encapsulated in a method of data objects, fifth program instructions including twelfth program instructions for identifying the data object that represents a component instance the user interacted with and for calling said method of said data object as necessary.

111. (withdrawn) The system of claim 110, further including thirteenth instructions for deciding based on said data to include more than one instance of a component into the generated document.

112. (withdrawn) The system of claim 109, further comprising twelfth program instructions for assigning a unique identifier to the first component instance, for associating the unique identifier with the data object, and for including the unique

identifier into the generated document, said twelfth program instructions for execution while dynamically generating a document .

113. (withdrawn) The system of claim 112, wherein fifth program instructions analyze said data for unique identifiers and include thirteenth instructions for identifying the associated data object.

114. (currently amended) A system for displaying dynamically generated documents in a data network coupling a server computer to a client computer, wherein the client computer has a first software program including first program instructions for generating a request to obtain at least one generated document from the server computer and for displaying the generated document on a display device, comprising:

a plurality of components having instructions for execution on the server computer, at least one of the components including first features adapted to cooperate with an editor in dynamically editing said component and second program instructions to generate browser code, and

a ~~third~~ program having instructions on the server for~~, based on the data contained in a request initiated by the client computer,~~ dynamically generating generated documents for transfer to the client computer based on the data contained in a request initiated by the client computer, thereby ~~calling~~ using second program instructions of selected components.

115. (previously amended) The system of claim 128 wherein first features include fourth program instructions for passing information to the editor.

116. (currently amended) The system of claim 115 wherein at least part of said information is collected during execution of selected components on the server computer.

117. (currently amended) The system of claim 115 wherein said information is transmitted from the server computer to the client computer.

118. (previously amended) The system of claim 115 wherein said information includes attribute values of said component.

119. (previously amended) The system of claim 128 wherein first features include fifth instructions that display additional editing features of the component during editing.

120. (previously added) The system of claim 119 wherein said editing features include handles.

121. (previously amended) The system of claim 128 wherein first features include an extension for use by the editor, said extension for enabling editing of an attribute value of the components .

122. (currently amended) The system of claim 121 wherein said extension enables display of a page for editing the ~~components~~ attribute values of the components.

123. (previously amended) The system of claim 128 wherein at least one component is denoted on at least one document template using tag syntax, whereby the tag name identifies a component kind.

124. (previously amended) The system of claim 114 containing at least one component wherein second program instructions are used to generate browser code for displaying the component during editing and during normal use.

125. (currently amended) A method for dynamically editing an application that is built using components and that operates by generating documents comprising the steps of:

running at least part of the application, thereby executing selected components and generating a generated document,

displaying a view of the generated document,

selecting a component by clicking on selected portions of said view,

identifying the selected component in the source code of the application, and

initiating a modification function modifying the source code of the application.

126. (currently amended) The method of claim 125 wherein the running step and the displaying step are repeated after ~~applying a~~ initiating the modification function.

127. (previously added) The method of claim 125 further comprising collecting edit information for use by the identifying step.

128. (previously added) The system of claim 114 additionally comprising a plurality of document templates with components denoted thereon, whereby the browser code generated by the components can vary for different requests of the same document template.